TPEP, Interpretations Process, and TCSEC Overview

This module describes the three phases of the Trusted Product Evaluation Program (TPEP), explains the process by which the TCSEC is interpreted within the scope of the TPEP, and provides a detailed overview of the TCSEC trust requirements.

Module Learning Objectives

The material in this module builds on the information presented in Module1. Upon completion of this module, the student should:

1. Understand the TPEP.
2. Understand the TCSEC Interpretations Process.
3. Understand the contents of the TCSEC.
4. Be aware of an example approach for mapping TCSEC requirements into a typical software development lifecycle.

Overview

Evaluation of computer products against the TCSEC is performed through NSA under TPEP. TPEP consists of three distinct phases encompassing the preparation of a product for evaluation, the evaluation of a product, andthe maintenance of a product's rating. This program is described in the following sections.

The general requirements contained within the TCSEC are often applied to specific computer product implementations using “TCSECInterpretations .” The process for the creation of these Interpretations is described withinthis module.

Finally, the TCSEC requirements, introduced in Module 1, are addressed in greater detail. This discussion is intended to provide an understandingof some of the subtleties of the requirements.

The Trusted Product Evaluation Program

TPEP focuses on the security features and assurances of commercially produced trusted products which include operating systems, networks and/or network components, and application products such as database management systems. TPEP consists of three phases: the Pre-Evaluation (advice)Phase , the Evaluation Phase, and the Rating Maintenance Phase (RAMP).

Pre-Evaluation Phase

Certain activities must be performed prior to starting an evaluation ofa trusted product. These activities are not part of the evaluation butrather are activities to ensure that the product and its associated evidence areready for the Evaluation Phase to begin. These activities include the preparation of a Product Proposal by the vendor, one or more Technical Assessments (TA) conducted by NSA as coordinated with the vendor, and at least one Intensive Preliminary Technical Review (IPTR). Each of these activities is described in more depth in the following paragraphs.

Product Proposal

After initial contact, NSA will provide the vendor with guidelines and requirements that instruct the vendor in the correct preparation of aProduct Proposal. NSA will review the vendor's product proposal to: ascertain ifit is likely to meet the TCSEC requirements, determine the potential market for the proposed product, and determine the readiness for an evaluation. If NSA, based on the proposal review, would like to consider a product for evaluation, the vendor is informed that the product is a candidate for evaluation.

Technical Assessments

Before evaluation can begin, the vendor must demonstrate readiness. During the vendor's preparation for evaluation, NSA representatives will make technical assessments of the vendor's progress. A TA will generally be an effort by a small team of NSA evaluators who meet with the vendor to assess ata high level, the work that needs to be completed before an IPTR can be performed. These assessments will vary in scope, level of effort, length, and evaluator resources based on the product complexity, estimated vendor preparedness, and targeted trust class.

To assist the vendor with their preparation for evaluation, NSA may recommend that the vendor seek outside advice or NSA advice, depending on the product's complexity, trust class and evaluator resources available at NSA. The first TA will normally help to determine how much advice is required and how long the advice should continue. When NSA provides advice, resources are prioritized according to trust class, the type of technology and the priority of the product for the DoD and the Intelligence Community. Products targeted at higher classes receive priority over products at a lower class, and products developed using a new and innovative technology receive a higher priority than products of outdated technology. Whether or not NSA can provide advice, vendors are always encouraged to seek computer security expertise from commercial sources.

Intensive Preliminary Technical Review

The IPTR is a two-week, in-depth analysis by NSA evaluators of thecurrent state of the product and its associated evidence to ensure the vendor's readiness to undergo evaluation. The IPTR team will not and cannotevaluate the product. The purpose of the IPTR is to accurately assess completenessof both the product and the evaluation evidence before transitioning intothe Evaluation Phase. If the IPTR recommends the product begin evaluation and provided adequate NSA evaluator resources exist, NSA will assign an evaluation team and the evaluation can begin. If the IPTR team does not believe the product is ready for evaluation, the reasons will becommunicated to the vendor and another TA or IPTR will be scheduled at a later, mutually agreed upon time, to determine if the deficiencies have been corrected. In order for a product to move into evaluation, at least one IPTR will have been held.

During an IPTR, the vendor is expected to present the product andexplain how the product meets each requirement for the candidate trust class in the TCSEC. Each IPTR team will write a report based on the vendor presentation

and documentation. The report includes an estimate of the readiness of the product, the required documentation, and a draft schedule for the evaluation. Once all of the required evidence exists, NSA and the vendor sign an Evaluation Agreement, and a team is assigned to the project.

Evaluation Phase

The Evaluation Phase begins with the vendor providing the evaluationteam with system-level, developer-oriented training for the vendor's product. Training is followed by a comprehensive review of the product design.The team performs security analysis of the product design including both hardware and software components of the product. The team reviews the completed design, user, test and RAMP documentation. The team uses the information gathered during design analysis to write an Initial Product AssessmentReport (IPAR), which is presented to an IPAR Technical Review Board (TRB). The IPAR TRB presentation: (a) demonstrates that the evaluation teamfully understands the product, and (b) assures consistency of ratingsamong evaluations. The team also briefs the TRB on the vendor's plans for testing the product. The evaluation team then performs security testing on theproduct at a site provided by the vendor. The team updates the IPAR to include testing results, and the name of the report changes to the Final Evaluation Report (FER). The evaluation team presents testing results at the Final TRB, the TRB makes its recommendations to NSA management, and NSA makes the final decision as to entry on the Evaluated Products List (EPL).

Rating Maintenance Phase

RAMP provides a mechanism for a vendor to maintain the TCSEC rating of a product throughout its life cycle. During RAMP, the vendor works with the NSA Technical Point of Contact (TPOC) to discuss possible changes to an evaluated product. The Vendor Security Analyst (VSA) performs a security analysis of the product changes as they occur. For C2 or B1 products, where the change analysis is entirely the responsibility of the vendor, the vendor describes all security-relevant modifications to a RAMP TRB, which makes a recommendation to NSA management on the continuation of a rating for the product. For higher trust class products (B2 and above), there is an advance analysis of the proposed changes and possible NSA resources applied to change analysis or penetration testing. A TRB recommendation is also made at the higher trust classes. If changes are approved, the original rating is applied to the current version of the product, and an entry in the EPL is made to document the maintenance of the rating. The FER is updated by the vendor. RAMP is mandatory for all products involved in the TPEP. RAMP is described in detail in Module 16.

TCSEC Interpretation Process

This section discusses the TCSEC Interpretation process. The existing Interpretations are discussed throughout this course within the moduleto which they apply.

One of the primary concerns of the Office of Product Evaluations andT echnical Guidelines is the uniform, consistent application of the TCSEC. The wide

physical distribution of evaluators and vendors makes this aparticularly difficult problem to solve.

Another concern is the independent nature of evaluation teams. Evaluations are performed by groups of evaluators working as teams. Every evaluator can not be on every team, so communication must be explicitly addressed. Ifone team tells a vendor that “X” is “OK” and another team tellsanother vendor that “X” is “NOT OK”, then the teams must be able to show why the situation is valid.

The other big concern is that the TCSEC is a high-level document. It avoided specific implementation detail in an effort to address allimplementations . In other words, team members have to make decisions on whether or not an implementation meets a given requirement.

Many of the concerns above can be addressed by improving communication channels and improving the evaluation community's understanding of the TCSEC itself. The Interpretation process helps do this by improving inter- and intra-team communication. This, in turn, helps improve the evaluation community's understanding of the TCSEC. In addition, the Interpretation process helps communicate this understanding out to the general computer security community.

The net effect of the Interpretation process is that it builds a body ofcase la w. A specific emphasis has been placed on streamlining the process toallow Interpretations to be created in a timely manner, yet still address the concerns noted above.

Support Mechanisms

The Interpretation process makes significant use of the “forum”utility on the NSA's Dockmaster computer system. Dockmaster is a Multics system operated by NSA for the purpose of facilitating discussion regarding computersecurity . One feature of the Multics system is a bulletin board mechanism known as “forum.” The forum mechanism supports many “meetings” (i.e., bulletin boards), where each meeting is devoted to a particular topic and explicitly defined audience.

For each NSA evaluation, two forum meetings are used to facilitate communication between team members and between the team and the vendor. The audience of one meeting is limited to the evaluation team membersonly . The other is limited to evaluation team members and vendor members only. Use of these meetings provides a written record of the issuesencountered during the evaluation. This record helps make these issues available to all the participants in a timely manner.

When an issue arises, it is discussed by the evaluation team and the vendor's personnel. If the team is able to reach a decision, it is documented on the team/ vendor meeting. In addition, it is posted to another meeting called **`Decisions`**. Access to this meeting is limited to evaluators since the information may be proprietary. Another meeting, **`VendorDecisions`**, exists to allow vendors to post non-proprietary decisions that the vendor believes should be knownby the general public.

The **Decisions** meeting allows one team to find out if some other team has already addressed a similar issue. Using the **Decisions** meeting encourages cross-fertilization of evaluation teams without having to give every evaluator access to every evaluation meeting.

Most issues are settled very quickly using the meetings noted above. Sometimes, however, a team can not reach a decision. In that situation, another meeting is used to discuss the issue. This meeting is called **Interpretations**.

The **Interpretations** meeting is intended to encourage and facilitate discussion among all evaluators on specific Interpretations of theTCSEC . Access is limited to current evaluators and a few other people who arecovered by non-disclosure agreements. The restricted access is required since proprietary issues are being discussed freely. Most issues discussed in this meeting are started because a team was not able to provide a clear answer to a vendor and wants input from the rest of the evaluation community. Sometimes this meeting is also used to voice disagreement with issues in thedecisions meeting, although this is very rare.

So far, the forum meetings which have been described have enabled evaluators to remain current of issues and to contribute to efforts which are not their primary duty. This does little to help people outside the evaluation community and it is easy to have the evaluators become “isolated.” Another meeting was created to address these problems. It is called **Criteria-Discussion**.

As the name suggests, the **Criteria-Discussion** meeting is to be used to discuss non-proprietary issues with the entire computer securitycommunity . Issues may be brought up by anyone and discussed by anyone. In addition, issues which are not proprietary are moved from the **Interpretations** meeting to this meeting. This move may result in parallel discussions on these meetings, but it allows input from the general community and it protects proprietary information. The text of pending Interpretations must beposted on the **Criteria-Discussion** meeting and left for comment before they can officially become Interpretations. This procedure allows the general community to be aware of what is happening before it actually takes place.

Final Acceptance

It should be noted that the hierarchy implied by the discussion of these meetings is not an intrinsic property of the forum utility. It requires activity on the part of all users and active management by the NSA. In particular, the Chief Evaluator is responsible for ensuring that outstanding issues are brought to closure.

Once an issue has been discussed and an Interpretation posted forcomment, the Chief Evaluator convenes a formal Interpretation meeting which is open to all evaluators. The views of the evaluators are considered along with any views from other people. An attempt is made to reach a consensus position on the Interpretation. An accepted Interpretation is prepared for concurrenceand signature by the Chief, Product Evaluations and Chief, Office of Product Evaluations and Technical Guidelines.

Once the Interpretation is signed, it is circulated for comment withinthe NSA. If no comments are received within two weeks, concurrence is assumed. Comments from other NSA offices will be addressed and incorporated intothe Interpretation if appropriate. Formal acceptance of an Interpretation is ultimately the responsibility of the Chief, Product Evaluations and Technical Guidelines.

Accepted Interpretations are posted on the **Announce** meeting. These Interpretations have the same authority as the TCSEC requirements, and vendors are expected to abide by all accepted Interpretations in effect atthe time the product enters the Evaluation Phase. Interpretations accepted while a product is in the Evaluation Phase will be handled on a case by case basis.

The current TCSEC Interpretations are as follows:

I-0001	Delayed enforcement of authorization change
I-0002	Delayed revocation of DAC access
I-0003	Access validation after object label change
I-0004	Enforcement of audit settings consistent with protection goals
I-0005	Action for audit log overflow
I-0006	Audit of user-id for invalid login
I-0007	Assigning device level range
I-0020	DAC authority for assignment
I-0022	One set of banner pages around multiple outputs
I-0039	Multilevel printers and page labeling
I-0040	Requirements for overwrite label capability
I-0041	Object reuse applies to all system resources
I-0043	Auditing use of unnamed pipe
I-0046	Detailed audit record structure
I-0069	Flexibility in packaging TFM
I-0073	OK to audit decision regardless of whether action completed
I-0084	Audit least disruptive action
I-0096	Blanking passwords
I-0144	Availability of diagnostics
I-0170	Functional tests required for object reuse
I-0172	Audit of imminent security violations
I-0192	Interface manuals as design documentation
I-0193	Standard system books as design documentation
I-0222	Passwords not acceptable for DAC
I-0239	Subject access revocation after change in user clearance
I-0240	Passwords may be used for card input
I-0244	Flexibility in packaging SFUG
I-0253	Default page marking format
I-0254	UNIX-style manual pages as DTLS
I-0275	Single-level printers and page labeling
I-0285	CM comparison source or object?
I-0286	Auditing unadvertised TCB interfaces
I-0288	Actions allowed before I&A
I-0312	Set-ID and the DAC requirement

C1-CI-01-83	Security Testing
C1-CI-02-83	Identification and Authentication
C1-CI-01-84	Audit
C1-CI-02-84	Security Testing
C1-CI-04-84	Audit
C1-CI-05-84	Exportation to Multilevel Devices
C1-CI-06-84	Discretionary Access Control
C1-CI-07-84	Audit
C1-CI-01-85	Device Labels
C1-CI-02-85	Audit
C1-CI-03-85	Discretionary Access Control
C1-CI-04-85	System Architecture
C1-CI-01-86	Discretionary Access Control
C1-CI-02-86	Server
C1-CI-03-86	DAC by Default
C1-CI-04-86	Operator Log-on
C1-CI-01-87	FTLS Accuracy
C1-CI-02-87	Audit
C1-CI-01-88	Exportation of Labels
C1-CI-01-89	Audit
C1-CI-02-89	Audit
C1-CI-03-89	DAC Public Objects

TCSEC Requirements

Module 1 briefly highlighted the fundamental differences between theTCSEC classes ranging from C1 to A1. This module builds on that presentation by discussing in more detail the functional and assurance requirements ofthe TCSEC by requirement. These added insights into the requirements enablea clearer understanding of the process by which specific implementations of those requirements are validated. In the following discussions, insights provided for a specific TCSEC class are also applicable to higher TCSEC classes unless otherwise stated.

Discretionary Access Control

Need-to-know control of information is implemented in one form or anotherin nearly every system on the market. Some implementations areoperationally awkward (e.g., file passwords), while some are quite flexible (e.g., access control lists). Discretionary access control (DAC) is fundamentally aweak property because while the user may have control over who has access to the object, he has no control over who has access to copies of the data. Therefore, a Trojan horse attack may be successfully mounted against a DAC system. However, DAC enforced need-to-know is an important mechanism since national security policy has two fundamental requirements -- proper security clearance and the need-to-know principle. New DAC requirements are introduced at the following classes.

C1: Basic DAC requirement - nominal user-controlled sharing of data is provided by the system. The C1 requirement implements a very basic form of DAC. Password mechanisms (as the only means of

protection) are sufficient only at this class. A password is not a very robust mechanism for protecting files since ANYONE who knows (or guesses, steals) the password will be able to get access to the file. The system may not be able to identify each individual using the system. As a result, the "owner" of the information contained in that file often has no way of knowing that only "authorized" users are using the file.

C2: Requires the ability to specify named individuals who get access to a particular data object, rather than relying merely on groups which might be based on the specific terminal being used or the login time of day. This has an influence on several other requirements, especially the "Identification and Authentication" (I&A) requirement. (See the I&A discussion on group accounts and passwords). SOME group mechanisms are allowed here (at C2), but NOT ALL of the ones for C1. Specifically, the group mechanism must still preserve the identity of the individual.

In addition, the C2 requirement contains words which limit the propagation of access rights. This part of the criterion requires that the mechanism by which a user is granted access to an object must be completely self-contained within the DAC mechanism. In other words, it must not be possible for a user to gain access to an object through an exchange of information or the passing of some attribute outside of the control of the TCB (e.g., file access protection using passwords). In that case, the TCB does not truly control access to the objects and would not pass C2 DAC requirements. Most systems meet this requirement within the fundamental limitation of discretionary controls. As noted above, the authorized user may not know whether another user has copied information into another object and allowed others to see that information. That, however, is a fundamental weakness of the DAC mechanism and not the focus of this requirement.

B3: The requirements are strengthened. The system must supply the equivalent of a user-controlled access control list (ACL) suchthat the individual can determine who can access data and in what modes they can access it without appealing to a system security officer or outside individual. The ability to explicitly deny access to an individual is also required. Authority and responsibility for controlling need-to-know is placed squarely on the individual user.

Now, the system must be able to deny (as well as allow) access based on individual identity. This is an implicit requirement for an ACL since an authorized user must be able to determine who can access data and in what modes they can access it. Other implementations are possible, but the most familiar is the ACL.

In all the classes, the meaning and semantics of the access modes must be specified. In other words, "write" access may imply (an unexpected) "read" access because of the way the hardware

performs the action. In and of itself, this may be acceptable. However, when combined with other features or naive design expectations, the results may be surprising.

Another point worthy of special note is that some systems may not be able to immediately revoke current access to an object, even though an authorized user may change the object's ACL. While it is desirable for a TCB to support immediate revocation of access, it is often too costly to perform an access control check each time a subject reads or writes an object. Thus, immediate revocation of a subject's DAC permissions is not required. In cases where immediate revocation is not possible, subjects already accessing an object may continue to use the object (e.g., using a descriptor to the object). Once the subject releases the object (e.g., givesup the descriptor to the object), the subject can't get it back. However, until the subject releases the object it may not be possible for the TCB to revoke access to the object. In such a system, about the only way to really remove all accesses to an object is to create a new object, containing the information with the proper access control entries, and delete the old object. That way all subjects are forced to go back though the access control mechanism. This sort of detail should be noted in the system's Trusted Facility Manual (TFM).

Labels

Labels are a significant distinguishing characteristic which separate division A and B products from division C and D products. Labels in the TCSEC represent sensitivity for restricting disclosure. Sensitivity labels have two parts -- a hierarchical *level* and a non-hierarchical *category set*. Labels in a TCB may be explicit (e.g., bits directly associated with data within a disk file's directory entry) or implicit (e.g., a CPU register implicitly labeled by the TCB based on what process is currently running), whichever is appropriate for the object or resource. The guidance on configuring mandatory access control (MAC) features (page 83 of the TCSEC) says one would normally expect to see at least 16 levels and 64 categories supported in a sensitivity label.These numbers are for guidance purposes only. In some applications, the recommended number of categories is woefully inadequate. The vendor should have a good understanding of their customers' needs, and should build their product accordingly.

A label has both an internal and an external representation. Theinternal representation may be a string of bits. The external representation is a human-readable expansion of the internal representation. The systemadministrator or security administrator should be responsible for specifying theexpanded form. This is discussed more in the section on "Labeling HumanReadable Output" of this module.

A TCB cannot control the flow of classified or other sensitive information unless it knows what information is classified at what level. Labels supply the classification of information, which is then compared with the user's clearance information. The user may specify a clearance at login (i.e., a requested session

label), which is then verified against the clearance information maintained by the system. The TCB controls the flow of information using theselabels . A Trojan horse may be able to circumvent the discretionary controls but it cannot circumvent system enforcement of security based on labels (i.e., a Trojan horse cannot downgrade or declassify information) without using some type of covert channel. The following requirements on labels are introduced at theindicated TCSEC classes.

B1: Subjects are labeled with their clearance to access data, and objects are labeled with the classification of their data.

B2: Labels are required for all objects that are accessible (directly or indirectly) by subjects external to the TCB. If something is "visible" at the TCB interface it must be labeled. For example, internal data structures can be modified which cannot be accessed directly by the user; however, the effects of the modification can be detected. This requirement includes information in read-only memory.

Label Integrity

In addition to using labels, there is a requirement that they be accurate. This requirement is known as label integrity, and it must be applied to all labels in the system. Printer spoofing and terminal labeling are issues here.

B1: Labels must be accurate in their representation of clearances and classifications and must be unforgeable by any unauthorized source. When labels are exported from the system, they must be unambiguously associated with the information. This includes unambiguous and unforgeable human-readable labels applied to printouts and terminal windows.

Mandatory Access Control

Through MAC, the TCB controls use of information within objects, not just access to objects. In other words, just because a subject has potential access to files does not imply that the subject is permitted to transferinformation between them in an arbitrary fashion. MAC prevents highly classified information from being transferred into an object labeled with a lower classification. This is accomplished by labeling subjects and objects, then controlling access based on these labels.

MAC is based on sensitivity labels. The rules used by a TCB to enforce MAC are based on a dominance relation between sensitivity labels. The TCSEC MAC requirements define the dominance relation that is to be used by a TCB. That is, one sensitivity label is said to dominate another if the following ruleis true:

> A dominates B if the hierarchical classification of A is greater than or equal to the hierarchical classification of B, and the non-hierarchical categories of A include all of the non-hierarchical categories of B.

The Bell-LaPadula model [Bell76] identified the following MAC rules, which the TCB must follow when granting access to objects:

Read access is granted only if the sensitivity label of the subject dominates the sensitivity label of the object.

Write access is granted only if the sensitivity label of the object dominates the sensitivity label of the subject.

Read/write access is only granted if the sensitivity label of thesubject equals the sensitivity label of the object.

B1: All subjects and objects under control of the TCB must be labeled and subject to the MAC rules. (Note that the B1 “System Architecture” requirement allows parts of the system to be “subsetted out” of the TCB.)

B2: The MAC rules must be applied to all accesses to objects that are accessible (directly or indirectly) by subjects external to theTCB. In other words, if something is visible at the TCB interface, it is labeled and the access control mechanism uses those labels. (Note that this requirement is tied to the change in the “System Architecture” requirement at B2.)

Exportation of Labeled Information

This requirement deals with how information is labeled as it enters, moves within, and leaves the TCB. All communication channels must be designated as either single-level or multilevel. Any change in this designation must be done manually and must be auditable.

The distinction between a single-level device and a multilevel devicesis whether the device is trusted to maintain the sensitivity label of datawhile the data is within the device. A multilevel device is trusted to correctly maintain the sensitivity labels of all the objects stored on the device. A single-level device does not separate objects on the basis of sensitivity labels. Objects on the single-level device are implicitly labeled by the TCB with the label ofthe device. For example, a given terminal may be used for logins from various classifications; however, each login has a single classification. Such a terminal is considered to be a single-level device, despite the fact that the terminal may operate at a different sensitivity label each time it is used by a user to login. The classification at which the user login occurs is considered the device's current label.

B1: The current label of a single-level device must be maintained and available for use in mandatory access control determinations. (The “Device Labels” requirement strengthens this requirement at B2.) Also, any change in the current label(s) of anysingle-level device must be audited.

Exportation to Multilevel Devices

A multilevel device must maintain labels. The device must be responsible for the association of labels and data. Typical examples of such devices are disk drives and network interfaces that support the labeling of network traffic.

To be a multilevel device, the TCB must be able to reliably send and receive a label to/from a multilevel device while the device must keep the label associated with the data to which it applies. This requirement simply says, "If you want it to be a multilevel device, you have to use it that way." That is, a device capable of multilevel service may be designated for use as a single-level device. If that is the case, the TCB must treat it as a single-level device.

For a device to be single-level, the label associated with the data mustbe maintained by the TCB, not the device. This is not the same as multilevel device that contains data having only one sensitivity label. If the label is maintained by the device and the TCB will make decisions based on thatlabel, then the device must be considered multilevel.

B1: Multilevel communication protocols must include the labels of data transmitted. The association of label and data on multilevel media must be unambiguous.

Exportation to Single-Level Devices

A single-level device does not maintain a label (or the TCB does notuse the label provided by the device). The TCB must keep a label which is to be associated with data coming from/going to a single-level device. Some single-level devices, over time, may actually process data with different labels (e.g., a terminal used to login users at various sensitivity labels one at atime). However, these devices are single-level because at any given time they are being used by the TCB to process a single level of data (the device'scurrent label), and at no time is the TCB relying upon the device to maintainlabels .

The system probably will not be able to properly set this label withouthelp from an authorized user. This criterion requires the label to be set to what an authorized user specifies, in a reliable manner.

B1: The TCB must ensure that the user and/or the TCB can reliably communicate the label of information imported from or exported to a single-level device.

Labeling Human-Readable Output

The system administrator must be able to specify printable labels. Any human-readable output (e.g., line printer output) must have labels at the beginning and end of each print job, and must have the capability to label the top and bottom of each page of the job. The TCSEC specifies that the TCB be capable of printing labels at the top and bottom of each page, not that it must. Therefore, a TCB may provide the capability for labels not to be printed on each page of a job. Instructing a TCB not to print labels on each page of a job is referred to as "overriding page labels." Any override of page labels is an auditable event. The ability to override page labels does not NEED to be a privilege, but it MUST be audited when used. Note that banner pages (i.e., the first and last pages of a print job) must always be labeled and protected from spoofing.

Given a TCB that supports the labeling of data at a page level as opposedto labeling of a file, that system's printed output could be multilevel. In such a system, the page labels may be different from each other because they should

reflect the sensitivity of the highest information on the page. The banner pages, however, must list the highest sensitivity of data in the print job. As a specific example, consider an output job with three pages. The sensitivity labels of each page may be Level3{Category0,Category1}, Level2{Category0,Category2}, and Level1{Category0}. The banner labels must then be Level3 {Category0, Category1,Category2}.

B1: The system administrator must be able to specify printable labels. Any human-readable output (i.e., line printer-type output) must have labels at the beginning and end of each print job, and must have the ability to label the top and bottom of each page of the job.

Terminals can be implemented as either single or multilevel devices. If they are single-level, the user must be notified of any label changes. If they are multilevel, the labels must be displayed appropriately (e.g., unforgeable labels in each window identifying the session label of that window).

Subject Sensitivity Labels

A sensitivity label must be associated with each subject (e.g., user process). The subject's sensitivity label must be dominated by the subject's clearance maintained by the TCB.

B2: The TCB must inform the user immediately of any changes in his/her current sensitivity label. The user must be able to request and be informed of his/her current sensitivity label.

Device Labels

Device labels allow an administrator to control information flow to andfrom each device. They could also be used to enforce security controls on the physical environment in which the device will be used. For example, device labels could be used to ensure that a printer residing in the computer room couldprint all levels of information, while a printer located in another part of thebuilding (less physically protected) could be restricted to non-sensitiveinformation only.

B2: The TCB must maintain minimum and maximum sensitivity labels for attached physical devices. This includes both single-level and multilevel devices. These sensitivity labels define the range over which the device may operate.

For a multilevel device, the range defines the sensitivity labels of data that may be read and written on the device. That is, the sensitivity label assigned to any information read from a device must dominate the device's minimum sensitivity label. The device's maximum sensitivity label must dominatethe sensitivity label of any information written to the device.

The need for a range of operation is not as obvious for single-leveldevices . As mentioned earlier in this module,it is possible for a single-level device to operate at different sensitivity labels at various times (e.g., one label per login session for a terminal). Given this possibility, the device label associated with a single-level device is used to selecti the sensitivity label at which the device will operate. That is, the single-level device may be assigned a sensitivity label

that dominates the minimum sensitivity label and is dominated by the maximum sensitivity label.

Identification and Authentication

The act of *identification* tells the TCB the name of the user before the user does anything on the system. The act of *authentication* proves to the TCB that the user really is the person that was identified. Both identification and authentication (I&A) are essential for the proper application ofdiscretionary and mandatory controls by the TCB.

Passwords are a typical method of authentication. [PASS85] is a good reference for guidance on password mechanisms. Passwords are not the only means to provide authentication. Biometric devices (e.g., retinal scanners, fingerprint readers) are becoming more widely available. Several token-based I&A subsystems have been evaluated by the NSA and could augment, or serve as, the primary authentication mechanism. Like passwords, these approaches have different strengths and weaknesses.

C1: Users must identify themselves to the system before doing anything. The TCB must authenticate the user's identity and must protect authentication data from access by unauthorized users. Note that a “user” can be considered a group at this point. This does not satisfy the goal of individual accountability, at higher classes.

C2: The TCB must be able to uniquely identify each individual who uses the system.

B1: The TCB must maintain the clearances and authorizations of each user.

Trusted Path

Trusted path is a way to guarantee that the user is communicating directly with the TCB. It is one highly effective means of preventing Trojan horse attacks.

B2: Users must be able to verify communication with the TCB before login. This helps prevent spoofing, which could result in users unwittingly revealing their password.

B3: The trusted path is used anytime spoofing could result in a violation of the system's security policy. This is essential anytime a change is being made to either a TCB data base or the rules that the TCB uses for I&A or MAC. DAC is explicitly omitted from this list (see Interpretation C1-CI-01-86).

Object Reuse

Data that is valuable from a penetrator's point of view can be gleanedfrom garbology (the analysis of garbage thrown out by others). Incomputer systems , objects like buffers are usually allocated to a given user for a shortperiod of time before being used by another user. If any information is left over, the next user may be able obtain valuable information left by the previous user. The

original user may not even know that a loss has occurred. This is an extreme problem if the previous “user” was the operating system itself.

Some classic examples of poor object reuse include deleting only the pointer to a file (during a file delete operation) and reading beyond thelogical end of a file . In the first case, several common utilities can rebuild a directory entry to the information which still exists. In the second, a user is allocated memory, writes at the end of this allocated memory, then tries to read any information left in the allocated (but not overwritten) memory.

The object reuse requirement demands that the system avoid this sort of problem. No information (including encrypted forms of information) may be left over. The system typically writes zeros into the object when it is returnedto a “free pool” or when the object is removed from the pool.

C2: Any time a storage object is used from a free pool of unused storage objects, the TCB must ensure that the subject is not getting data for which the subject is not authorized. This requirement can be very hard to satisfy since many systems have complex memory management mechanisms. An evaluation team examines all the mechanisms a given system uses (including page swapping, etc.).

B3: Object reuse can be a major concern after a system crash. This concern is related to the “Trusted Recovery” requirement, and both object reuse and trusted recovery must be compatible and effective across crashes and other system discontinuities.

Audit

Auditing is used to determine “who did what” to the system. It hasseveral purposes, including monitoring file accesses and login attempts. The TCSEC audit requirement does not insist that audit records be created as aresult of system activity; instead, it demands that the system be CAPABLE of creating audit records reflecting system activities. The TCSEC audit requirement calls for an administratively controllable mechanism. Thus, each site may determine the appropriate audit related overhead that satisfies its needs.

C2: The TCB must be able to audit security-relevant user actions. The following are some examples of events that must be auditable (these are in no way a complete list):

Login (record the physical location of the user).
File open (record the name of the file).
Object deletion (record the name of the object).
Changes to DAC information.
Any operator and/or administrator actions.

It is acceptable to audit only initial object accesses (e.g., file open) if that is the only time an access control decision is performed. In general, an auditrecord should be generated to document the result of every access control decision.

Audit records must contain: the type of event, the date and time of theevent, the identity of the user who directly caused the event, and whether or notthe

requested action was performed. The mechanism should not record the value of a bad password. The TCB must let the system administrator choose the users whose actions are recorded in the audit trail or tools must beprovided to select a user's actions from the audit records.

B1: The list of auditable events must include any override of human readable output markings (e.g., page labels). The audit records should also contain object sensitivity labels whenever it is appropriate.

B2: Any events which can be used to exploit covert storage channels must be auditable by the TCB.

B3: The system must include the capability to monitor the accumulation of auditable events (e.g., denied login), issue real-time alerts to the security administrator, and take the least disruptive action to terminate the event(s) if violations continue (e.g., terminate login sequence or lock the user ID for some specified time interval). A formal Interpretation exists on this topic (C1-CI-02-87).

System Architecture

The system architecture requirement has a lot of impact on a system. At the lower classes (C1-B1), portions of the system may be “subsetted” out of the evaluated system. This subsetting also means that they can not be available to the general users. The ideal situation (or as ideal as subsetting can be) is to totally remove the subsetted portions from the system.

C1: The TCB must be self-protecting, and may be a defined subset of the system.

C2: The TCB must provide isolation of the resources to be protected so that they can be audited and are subject to the access control requirements.

B1: User processes are isolated from one another using (at least) address space controls.

B2: The TCB must control the entire system, enforce the principle of least privilege, and be internally structured into modules. The TCB user interface must be completely defined. All elements of the TCB must be identified. Therefore, rigorous application of good software engineering practices are necessary to address the majority of the architecture requirements. These topics are discussed in more detail in Module 7.

B3: The TCB must implement the reference monitor concept and be developed using structured development methods. The key aspect is that the TCB must be reduced to the minimum size by removal of non- security-relevant portions. The size of the TCB is inversely proportional to the ability to provide assurance that it is implemented correctly.

System Integrity

The system integrity requirement is intended to allow a systemadministrator to verify that the hardware and firmware portions of the TCB are operating correctly. This requirement is often satisfied by a suite of diagnostics.

C1: There must be a provision for testing the correct functioning of hardware and firmware elements of the TCB. The system administrator must be able to run these tests periodically. Many systems provide some tests which run at system power-up. Other systems require the assistance of vendor field engineers to execute the tests. Thus, diagnostics may be provided as part of the TCB or made available separately. That is, it is acceptable for an administrator to be required to call field engineers to have the tests executed. It is also acceptable to require the tests be executed in a “stand alone” environment.

Covert Channel Analysis

In the DAC requirement, it was noted that discretionary controls have a problem because Trojan horses can operate with a user's access rights/ privileges. Mandatory controls have a similar problem with covert channels. Covert channels are an unintentional and unavoidable side effect of sharing resources. Eliminating the dynamics of sharing defeats the purpose of having multilevel computers. There are two types of covert channels: covert storage channels and covert timing channels.

Covert storage channels use a storage object to send information (e.g., modulation of a “disk full” state). Timing channels use time as the modulated resource (e.g., modulation of system response time). Section 8 of the TCSEC recommends possible methods for dealing with covert channels depending on the bandwidth of the channel. The goal is to try and eliminate or reduce the bandwidth of covert channels. Auditing is important, especially when it is impossible to eliminate the channel. The bandwidth constraints shown within the TCSEC are based on the assumption that terminals transfer information at a rate of 100 bits per second. The TCSEC states:

> “It does not seem appropriate to call a computer system 'secure' if information can be compromised at a rate equal to the normal output rate of some commonly used device.”

Note that there are typically two or more subjects (e.g., processes) cooperating in order to effectively utilize a covert channel. One (or even both) of those processes may be cooperating because of a Trojan horse. In such cases, a user could be participating in the downgrading of information unknowingly.

The requirement for a covert channel analysis also ties into the real-time alert requirement for audit. The primitive event(s) (e.g., disk reads, disk writes, clock reads) used by a covert channel are often valid operations for a subject. In general, it is hard to tell a valid use of a primitive event from onewhic h is being used to deliberately signal information. Therefore, the primitive events identified as potential covert channel uses may need to be audited and subject to threshold controls and alarms in order to provide a throttle for acovert

channel. An operator or administrator would then be required to make a judgment based on the alarms and audit data.

B2: The system developer must conduct a thorough search for covert storage channels and determine the maximum bandwidth of each of these channels. The requirement is actually for more than simply a search for covert channels. Potential channels found during the analysis must be addressed to bring their bandwidth under control as much as possible. Discuss the results of a covert channel analysis with the NSA Technical Point of Contact.

B3: The search should also be made for covert timing channels. Also, note the tie to the "Audit" requirement that requires the least disruptive action to terminate the exploitation of the covert channel.

A1: Formal methods should be used in the covert channel analysis.

Design Specification and Verification

Models, both formal and informal, are effective ways to clearly and concisely describe system activity.

B1: A model (formal or informal) of the security policy supported by the TCB shall be maintained and shown to be consistent with its axioms. This is tied to the "Design Documentation" requirement.

B2: Unlike B1, a B2 model must be formal. It must be proven to be consistent with its axioms (the Bell-LaPadula model [Bell76] is an example of a formal model). Also, a Descriptive Top-Level Specification (DTLS) of the TCB shall be maintained. The DTLS describes the interface to the TCB and those aspects of the TCB that are visible at the interface. These documents must be maintained over the lifecycle of the system.

B3: A convincing argument must be given that the DTLS is consistent with the model. This is typically some form of mapping between model states, axioms and transitions to DTLS statements.

A1: The Formal Top-Level Specification (FTLS) must be shown to be an accurate description of the TCB interface, and a combination of formal and informal techniques must be used to show that the FTLS is consistent with the model. Manual or other mapping of the FTLS to the TCB source code shall be performed to provide evidence of correct implementation.

Design Documentation

Design documentation provides the rationale and supporting evidence forthe security of the design of the system. The descriptive specificationsare inc luded in this evidence, too. This material should not be something which is created just to satisfy the evaluation requirements. It helps provide the sort of information a new developer would need in order to support the system.

C1: The documentation includes the manufacturer's philosophy of protection. If the TCB consists of distinct modules, the design documentation describes the interfaces between these modules. Note that auditing at C2 is an interface, and is covered despite the "no additional requirement" in the requirements summary. The "philosophy of protection" forms the first movement toward having a model of the system.

B1: The documentation describes the security policy model and how the TCB implements it.

B2: The documentation describes how the TCB implements the reference monitor concept and how it enforces the principle of least privilege. It also describes the results of covert channel analysis.

The interfaces between the TCB modules shall be described. This is an influence from the "System Architecture" requirement.

B3: The TCB implementation is shown to be informally consistent with the DTLS. The DTLS elements shall correspond to TCB elements (informal).

A1: The design documentation also describes components that are strictly internal to the TCB. You need to specify this information to extend the value of the verification results to the actual implementation of the system.

Trusted Facility Management

Certain functions are of such a nature that, if they were to be abused, they could negate the protection provided by all of the security mechanisms. These include adding a new login ID to the system, specifying the clearance associated with each account, changing the sensitivity label of objects, controlling the collection of audit records and others. Therefore, the system must be able to differentiate between the administrative users who areallowed to perform those functions and the administrative users who are not.

While the people on the system administration staff (e.g., the operator and administrator) are considered trusted users, they do not require unlimited capabilities to adequately perform their duties. An operator whose duties include mounting tapes, monitoring printers, and halting the system, does not require the ability to add new users. Thus, greater assurance that a site is secure comes from greater control over these trusted users. A typical form of control is the definition of administrative roles each having a precisely defined scope of authority (e.g., an operator role, security officer role, system administrator role, an auditor role).

The emphasis is not on whether or not the person is trusted, but on howmuc h the subject (e.g., process) representing the person is constrained. The primary concern is what Trojan horse code can do executing on behalf of aprivileged user. Because of this concern, the constraints on administrative roles mustnot only include limits on the security-critical operations that can be performed, but also limits on the ordinary activities that can be performed. Bylimiting the

administrative roles to administrative actions, the possibility that such a user inadvertently executes a Trojan horse is reduced.

B2: The TCB must support separate operator and security administrator functions.

B3: The TCB must also support a separate role of security administrator, who performs only those functions directly related to security. If the administrator wants to perform normal user type activities, then he must have another user account for them. He can only assume this role after a distinct auditable action. This role is to be strictly limited in what it can do.

Trusted Recovery

After a system crash or a shutdown, it is vital that the system bebrought bac k up into a secure state. This requirement does not guarantee that a site will be able to recover data lost in a head crash, for instance. It does require the TCB to continue to protect the information it (still) controls.

B3: The system shall be secure even after a crash or shutdown. Data must be properly labeled (preventing MAC violations), free pools must be properly initialized (preventing object reuse violations), memory images must be properly protected (preventing disclosure of data from different address spaces), etc.

Configuration Management

During the design, development, and implementation of a system, theremust be some sort of control over the creation and modification of systemmodules to have assurance that the implementation faithfully adheres to the design specifications. While the TCSEC requires configuration management for B2, B3 and A1 class system only; products targeting other classes should reference the RAMP requirements.

Configuration management, in terms of the TCSEC requirements, is more than just source code version control. It includes control over the design documentation and test suites, analysis of proposed design changes, and continual security scrutiny, too. The assurance comes from protecting the system at the start, and ensuring any changes are analyzed for their effects upon security BEFORE they are introduced.

There is an implicit tie between writing new code and the need to testthat code . It is a good idea to address the testing issue while developing new code.

B2: Control must be maintained over changes to the TCB during development and maintenance of the TCB. Tools must be provided to generate a new TCB from source code, and to compare the new TCB with the old one to ensure that only the intended changes have been made.

A1: Control must be maintained over the TCB during the design phase as well as during the development and maintenance processes. Comparison tools must be maintained under strict

configuration control. The master copy of the TCB must be protected from unauthorized modification or destruction.

Trusted Distribution

A computer center must be sure that the system they are operating isidentical to the vendor's evaluated product. The trusted distribution requirementcalls for procedures to provide assurance that this is in fact the case.

A1: Procedures must be provided by the product vendor that demonstrate that the on-site copy of the TCB is identical to the master copy maintained by the vendor. For the purposes of this requirement the TCB includes hardware, firmware, and software updates. with the intended versions. This requirement also applies to system updates.

The on-site code may be source or binary. In either case, the goal is to show that it is really the proper code. No specific scheme is required because the various ways of doing business and customer demands can not be resolved with just one method. The vendor must convince the evaluation team that what they (the vendor) have does the job.

Security Features User's Guide

The Security Features User's Guide (SFUG) tells users how to use the security mechanisms of the system. It should provide an understanding of how to effectively use the I&A, DAC, and MAC mechanisms. The SFUG must also include warnings and instructions outlining good security practices.

C1: Documentation must describe the protection mechanisms provided by the TCB and how to make effective use of them.

Trusted Facility Manual

The TFM tells the system administrator how to setup the system so that itis secure and stays secure. It tells the administrator how to select the proper options such that the system is operated in a mode that meets the requirements of the TCSEC. If there are other modes, the TFM should clearly state their impact on security and include warnings as appropriate. This manual should also include any procedures the administrator should use during system operation to maintain security. If any of the hardware/ software features require administrator actions to complete the securityprotection, they should be thoroughly described.

C1: The manual tells how to run the system in a secure manner.

C2: The manual describes procedures for maintaining audit data.

B1: The manual describes the operator and administrator functions related to security.

B2: The manual tells what constitutes the TCB, how to modify it and securely generate a new TCB from source. Also, the separation of operator and administrator functions need to be described.

B3: The manual tells how to start-up and restore system operation securely. The security administrator role is described.

Test Documentation

This documentation is used by the evaluation team to assess the testing performed by the system developer. The preparation of functional tests is the responsibility of the vendor. This document describes those tests, how they are run, and how to properly interpret the results.

C1: The documentation describes results of security mechanisms' functional testing. This includes the procedures used to perform the tests.

B2: The documentation includes results of covert channel tests.

A1: The documentation includes results of the mapping between the FTLS and the TCB source code.

Security Testing

Security testing demonstrates that the system actually exists and thatit actually works the way it is supposed to. Note that this requirement is somewhat different from the “System Integrity” requirement,although some tests may help meet both requirements.

The distinction between functional security tests and penetration testsis more interesting at the “high” classes of the TCSEC (i.e., classes B2, B3 and A1). Functional security tests should be a part of the quality assuranceprogram for all products at all classes. At the “higher” classes, the “System Architecture” requirement has forced a modular design which is easier to analyze and test.

Penetration testing is required on systems targeted at TCSEC class B2, B3 or A1. Penetration testing is performed at these classes simply to increase assurance in the TCB. Penetration testing does not provide conclusive results; however, it can provide increased assurance. If a team is unable to penetrate a system, either the system is very good or the team is not. On the otherhand, if the team finds a lot of holes, the resulting assurance is not very high. The result which should be expected is that the system is penetrated.

The issue of correcting flaws is unclear. Whether the vendor is allowed to correct a flaw is currently at the judgment of the team (during the Evaluation Phase). Implementation errors (typographical errors, etc.) are fixed. When design errors are fixed, those fixes can impact other areas of thesystem.

C1: The security features of the TCB must be tested and found to work as claimed in the system documentation. A search must be conducted for “obvious” flaws. Obvious flaws are those that canbe identified from a through review of the user's documentation.

C2: Testing will include the audit protection mechanisms.

B1: All discovered flaws must be removed or neutralized, and the TCB must then be retested. One user should not be able to make the TCB deny service to other users.

B2: All discovered flaws must be corrected, and the TCB retested. The TCB must be relatively resistant to penetration.

B3: No design flaws and no more than a few correctable implementation flaws may be found.

A1: The implementation of the TCB is demonstrated to be consistent with the FTLS. The FTLS mapping to the source code is one possible source for penetration testing.

Summary of the Classes Revisited

The new features required in each class are listed as follows:

C1: Identification and authentication
Discretionary access control
Self-protection system
Documentation
Functional security testing

C2: Individual accountability
Audit trails
Object reuse requirements

B1: Labels
Nominal mandatory access controls
Informal security policy model

B2: "Real" mandatory access controls
Trusted path between user and TCB
Structured TCB
Configuration management
Attention to covert channels
Penetration testing

B3: TCB is based on a formal model
Trusted facility management
Trusted recovery
Substantial documentation:
* Model
* Descriptive top-level specification
* Design analysis
* Trusted facility manual
* Test results description
* Security features user's guide

A1: Formal top-level specification
Formal verification of the design
Stringent configuration control
Stringent distribution procedures
TCB is derived from FTLS
Documentation on:
* Hardware/software not included in FTLS
* Mapping between FTLS and source code

The TCSEC embodies principles for developing trusted systems, but it ignores the software development process. It is intended that existing development paradigms be adapted to meet the TCSEC requirements. Various milestones specific to trusted product development should be interspersed withthe normal milestones of system development. A detailed mapping of TCSEC and TPEP needs and reqirements to a model software development paradigm is described in [Benzel89]. Note that the use of DoD-STD 2167A [DSSD88] in [Benzel89] does not in any way imply the need or requirement for vendors to employ this particular development paradigm.

Relevant Trusted Product Evaluation Questionnaire Questions

None.

Required Readings

TCSEC85 National Computer Security Center, *Department of Defense Trusted Computer System Evaluation Criteria*, DoD 5200.28-STD, December 1985.

Sections 2.1, 2.2, 3.1, 3.2, 3.3 and 4.1 contain the requirements described throughout this module. These requirements are summarized in Appendix D.

Supplemental Readings

Bell76 Bell, D.E. and La Padula, L.J., *Secure Computer Systems: Unified Exposition and Multics Interpretation*, MTR-2997, Rev. 1, MITRE Corporation, Bedford, MA, March 1976.

INTERP94 National Computer Security Center, *The Interpreted TCSEC Requirements*, (quarterly).

The Interpretations are included with the materials for the module to which they apply. There are currently Interpretations for the following modules: Architecture, MAC, DAC, I&A, Audit, and Assurance.

ENVR85 DoD Computer Security Center, *Technical Rationale Behind CSC-STD-003-85: Guidance for Applying the DoD TCSEC in Specific Environments*, CSC-STD-004-85, 25 June 1985.

PASS85 National Computer Security Center, *DoD Password Management Guideline*, CSC-STD-002-85, April 1985.

Other Readings

Benzel89 Benzel, T.C.V., "Developing Trusted Systems using DoD-STD-2167A," *Proceedings of the 5th Annual Computer Security Applications Conference*, pp. 166-176, December 1989.

DSSD88 Department of Defense, *Military Standard: Defense System Software Development*, DOD-STD-2167A, 28 February 1988.